

THE DOMAIN REDUCTION METHOD: HIGH WAY REDUCTION IN THREE DIMENSIONS AND CONVERGENCE WITH INEXACT SOLVERS*

CRAIG C. DOUGLAS[†] AND JAN MANDEL[‡]

Abstract. We study a method for parallel solution of elliptic partial differential equations which decomposes the problem into a number of independent subproblems on subspaces of the underlying solution space. Using symmetries of the domain, we obtain up to 64 such subproblems for a 3 dimensional cube and the method reduces to a direct solver. In the general case, or when the subproblems are solved only approximately, the method becomes an iterative method or can be used as a preconditioner. Bounds on the resulting convergence factors and condition numbers are given.

1. Introduction. In this paper, we approximate the solution to the elliptic boundary value problem

$$(1.1) \quad \begin{cases} \mathcal{L}u = f & \text{in } \Omega \subset \mathbb{R}^d, \quad d > 0 \\ u = 0 & \text{on } \partial\Omega_D, \quad \text{where } \partial\Omega = \partial\Omega_D \cup \partial\Omega_N, \\ \frac{\partial}{\partial n}u = 0 & \text{on } \partial\Omega_N, \quad \text{and } \partial\Omega_D \cap \partial\Omega_N = \phi, \end{cases}$$

using a combination of domain decomposition, multigrid, and projection method techniques. We assume that (1.1) is well posed and will be discretized by a finite element, volume, or difference scheme. Typically,

$$\mathcal{L}u = - \sum_{i,j=1}^d \frac{\partial}{\partial x_i} \left(a_{ij}(x) \frac{\partial u}{\partial x_j} \right) + a_0(x)u.$$

* Chapter 9 in *Fourth Copper Mountain Conference on Multigrid Methods*, edited by J. Mandel, S. F. McCormick, J. E. Dendy, C. Farhat, G. Lonsdale, S. V. Parter, J. W. Ruge, and K. Stüben. Published by SIAM Books, Philadelphia, 1989, pp. 149–160.

[†] Mathematical Sciences Department, IBM Research Division, Thomas J. Watson Research Center, P. O. Box 218, Yorktown Heights, NY 10598.

[‡] Computational Mathematics Group, Department of Mathematics, University of Colorado at Denver, Denver, CO 80204. This research was partially sponsored by the National Science Foundation under grant DMS-8704169.

The domain reduction method uses properties of a partial differential equation to split the problem into several subproblems. Each subproblem should be solved in parallel using the fastest known solution method appropriate.

The basic method was first derived in [8] as a parallel multilevel method for linear systems of equations where smoothing was unnecessary. The applicability to shared or distributed memory, coarse or fine grained parallel computers was discussed informally in [9]. The elliptic partial differential equation case (with an emphasis on one and two dimensions) was analyzed in [10], and a technique for doubling the basic parallelism was introduced in [3].

In this paper, we first extend the idea in [3] to three dimensional problems, where up to an eight fold increase in the basic parallelism can be accomplished. We are once again interested in the case where smoothing is unnecessary, making this a projection method. We then extend the analysis in [10] to the case when subproblems are solved inexactly and for the case when the method is inherently iterative.

In §2, we define the algorithm as it pertains to this paper in a very abstract manner. We also define a simple 2^d way domain reduction when $\Omega = (-1, 1)^d$. In §3, we extend the eight way domain reduction for a 3 dimensional cube to nonobvious 60 and 64 way domain reductions. Finally, in §4 we show that if the subproblems are solved only approximately, then the method turns into an iterative method with the convergence factor at most equal to the convergence factor of the iterative method (or methods) used for solving the subproblems as a special case of a general result for methods reducing the problem to subproblems on subspaces.

2. The Direct Algorithm. We formulate (1.1) variationally as follows:

$$(2.1) \quad \text{find } u \in V \text{ such that } a(u, v) = f(v), \forall v \in V.$$

Typically, $V = V(\Omega)$ is the appropriate Sobolev space, and

$$a(u, v) = \int_{\Omega} a \nabla u \nabla v + a_0 uv dx \quad \text{and} \quad f(v) = \int_{\Omega} f v dx.$$

The method may be formulated quite abstractly:

ALGORITHM 1. *To solve (2.1), we find subspaces V_1, V_2, \dots, V_n such that*

$$(2.2) \quad V = \bigoplus_{i=1}^n V_i \quad \text{and} \quad V_i \perp V_j, \quad i \neq j,$$

in the $a(\cdot, \cdot)$ inner product. Then the solution is

$$u = \sum_{i=1}^n u_i,$$

where u_i is the solution of the subproblem

$$(2.3) \quad \text{find } u_i \in V_i \text{ such that } a(u_i, v) = f(v), \forall v \in V_i.$$

In implementation, full-rank restriction and prolongation operators are used to transfer information between the subspaces and V :

$$\mathcal{R}_i : V \rightarrow V_i \quad \text{and} \quad \mathcal{P}_i : V_i \rightarrow V.$$

Frequently, each prolongation operator is the adjoint of one of the restriction operators. Writing the problem (2.3) as $\mathcal{L}u = f$, we find that the corresponding operators for the subproblems are then defined in the Galerkin sense:

$$\mathcal{L}_i = \mathcal{R}_i \mathcal{L} \mathcal{P}_i.$$

Examples of several choices of restriction and prolongation operators, and the resulting \mathcal{L}_i , are given in [10].

We complete this section with a simple example where the domain reduction method uses symmetries in (1.1) to split the problem into several subproblems. For $x = (x_1, \dots, x_d) \in \Omega = (-1, 1)^d$, define

$$\mathcal{R}_{i_1, \dots, i_d} x = ((-1)^{i_1} x_1, \dots, (-1)^{i_d} x_d), \quad (i_1, \dots, i_d) \in \{0, 1\}^d.$$

We define the subspaces $V_{i_1, \dots, i_d} \subset V$ by

$$V_{i_1, \dots, i_d} = \{ u \in V : u \circ \mathcal{R}_{i_1, \dots, i_d} = u \}, \quad (i_1, \dots, i_d) \in \{0, 1\}^d.$$

The following lemma is immediate:

LEMMA 1. *It holds that*

- (a) $V = \bigoplus_{i_1, \dots, i_d=0}^1 V_{i_1, \dots, i_d}$, and
(b) $a(u, v) = 0$ if $u \in V_{i_1, \dots, i_d}$, $v \in V_{i'_1, \dots, i'_d}$, and $(i_1, \dots, i_d) \neq (i'_1, \dots, i'_d)$.

Therefore, we may split (2.1) into 2^d subproblems:

$$(2.4) \quad \text{find } u_{i_1, \dots, i_d} \in V_{i_1, \dots, i_d} \text{ such that } a(u_{i_1, \dots, i_d}, v) = f(v), \quad \forall v \in V_{i_1, \dots, i_d},$$

and recover the solution of (2.1) as

$$u = \sum_{i_1, \dots, i_d=0}^1 u_{i_1, \dots, i_d}.$$

Let $\tilde{\Omega} = (0, 1)^d$. For a function \tilde{v} on $\tilde{\Omega}$, define prolongation operators $\mathcal{P}_{i_1, \dots, i_d}, (i_1, \dots, i_d) \in \{0, 1\}^d$, by

$$(\mathcal{P}_{i_1, \dots, i_d} \tilde{v})(x) = (-1)^{i_1 i'_1 + \dots + i_d i'_d} \tilde{v}(\tilde{x}), \quad \text{for } x = Q_{i'_1, \dots, i'_d} \tilde{x}.$$

Then, (2.4) can be formulated as a problem on the domain $\tilde{\Omega}$ of the form

$$(2.5) \quad \text{find } \tilde{u}_{i_1, \dots, i_d} \in \tilde{V}_{i_1, \dots, i_d} \text{ such that } \tilde{a}(\tilde{u}_{i_1, \dots, i_d}, \tilde{v}) = f(\tilde{v}), \quad \forall \tilde{v} \in \tilde{V}_{i_1, \dots, i_d},$$

where

$$\tilde{V}_{i_1, \dots, i_d} = V_{i_1, \dots, i_d} \cap V(\tilde{\Omega}),$$

TABLE 1
Numerical Experiments on Encore Multimax for Poisson's Equation

DR = domain reduction, *MG* = multigrid, *SGE* = sparse Gaussian elimination

$$d = 2, \quad N = 64^2$$

Method	Processors	Time	Speedup Factor
<i>SGE</i>	1	492s	
<i>DR + SGE</i>	1	141s	1.00
<i>DR + SGE</i>	2	78s	1.81
<i>DR + SGE</i>	4	36s	3.92
<i>MG</i>	1	9s	
<i>DR + MG</i>	4	2.2s	

$$d = 3, \quad N = 30^3$$

Method	Processors	Time
<i>MG</i>	1	48s
<i>DR + MG</i>	8	6.2s

individual processor, there is virtually no communication, and we have an example of embarrassingly parallel computation. Experiments are currently underway to use more processors by parallelizing the multigrid solver.

When the subproblems are solved directly, the run time speedup is superlinear with respect to the time for solving the original problem similarly on a single processor. This is expected since the solve time, assuming a band solver, is N times the bandwidth (in this case, $N^{(d-1)/d}$), or $cN^{(2d-1)/d}$. The computation time for each subproblem is proportional to $c(N/2^d) \cdot (N/2^d)^{(d-1)/d}$. In our case, this is

dimensions	leading term in time expansion
d	$c \cdot 2^{1-2d} \cdot N^{(2d-1)/d}$
2	$(c/8) \cdot N^{3/2}$
3	$(c/32) \cdot N^{5/3}$

which is clearly shown in Table 2.1. Also, the storage used is reduced significantly due to the greatly reduced matrix bandwidth for the subproblems. The communication time is less than 1 percent of the solution time even for the moderate sized problems in Table 2.1.

When the subproblems are solved to the order of the truncation error using a multigrid solver, the time and storage is linear in the number of unknowns. For the problems tried to date, this is the method of choice. However, for some problems derived from parabolic partial differential equations, using an implicit time discretization scheme (e.g., Crank-Nicolson), a (parallel) preconditioned conjugate gradient method based on a predictor-corrector (or extrapolation) scheme is undoubtedly superior.

On fine grained parallel computers (e.g., Thinking Machines CM-2), where a fast Fourier transform like procedure can be applied, resulting in $O(\log N)$ run times,

domain reduction can reduce the run time to $O(\log(N/d))$. This is faster than any known method so far by a small, but quite significant constant.

3. High Way Domain Reduction. In this section, we construct a different eight way reduction of the 3 dimensional cube, based on the number of Neumann and Dirichlet boundary faces. This work is an extension of the two dimensional case in[3]. By translating the coordinates and composing simple operators, we extend the eight way domain reduction in §2 to a 60 or 64 way domain reduction. We note that even if we want to solve a variable coefficient problem, we can use the technique of this section as a very efficient preconditioner to the original problem.

While we push this technique to the extreme in this section, someone using these techniques should not feel obligated to do so. For example, it may be more convenient to use a 16 or 32 way domain reduction instead of a 64 way domain reduction, due to the number of processors on a machine or merely programming convenience.

This time, we produce problems on four distinctly shaped domains. When we restrict ourselves to subdomains with 45° and 90° angles only, we can produce a 60 way domain reduction where all but four of the subproblems have $N/64$ unknowns and four subproblems have $N/32$ unknowns. If we drop the angle condition, we can produce a 64 way domain reduction, where each subproblem has $N/64$ unknowns.

There are four distinct cases to consider, based on the number of Neumann boundary faces (zero, one, two, or three) described in (1.1) for $\Omega = (-1, 1)^3$. There are one, three, three, and one, respectively, of each. The case of zero Neumann boundary faces was covered in §2 (see Figure 3.1(a)).

Consider the case of one Neumann boundary face on $\partial\Omega$, say $x_2 = 1$. By folding the domain first about the x_1 -axis, then the x_2 -axis, and finally the diagonal with endpoints at $(.5, 1, x_3)$ and $(.5, .5, x_3)$, we are left with a wedge domain $\tilde{\Omega}^{0,1,0}$ with coordinates

$$\frac{1}{2} \leq x_1 \leq 1, \quad \frac{3}{2} - x_1 \leq x_2 \leq 1, \quad \text{and} \quad 0 \leq x_3 \leq 1$$

(see Figure 3.1(b)). Let $i_4, i_5, i_6 \in \{0, 1\}$ and $x = (x_1, x_2, x_3) \in \Omega$. Then the operator $\mathcal{P}_{i_4, i_5, i_6}^{(0,1,0)}$ is given by

$$\mathcal{P}_{i_4, i_5, i_6}^{(0,1,0)} x = ((-1)^{i_6}(\text{sgn}((-1)^{i_4} x_1) - (-1)^{i_4} x_1, (-1)^{i_6}(\text{sgn}((-1)^{i_5} x_2) - (-1)^{i_5} x_2, x_3),$$

where for $y \in \mathbb{R}$,

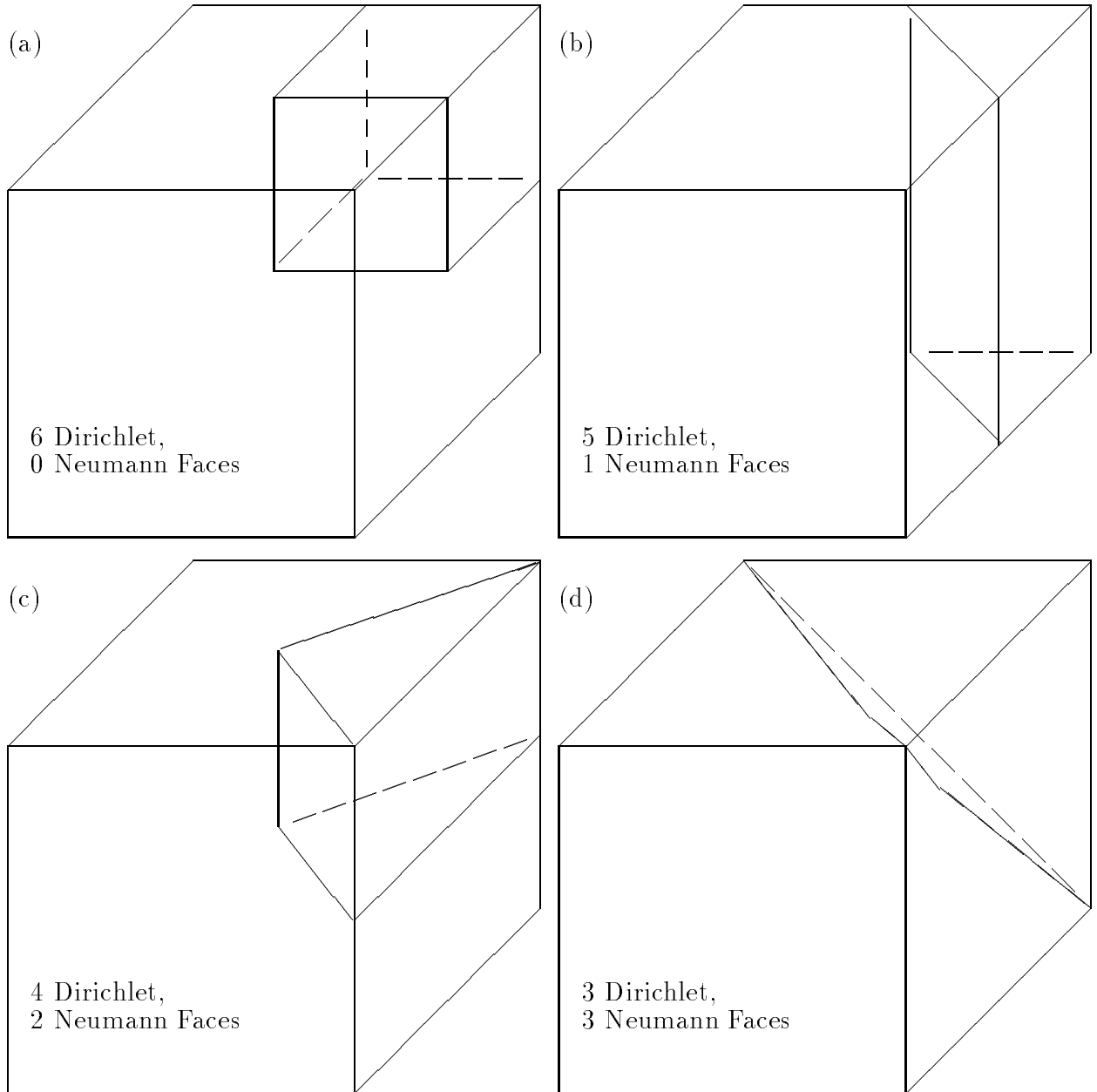
$$\text{sgn}(y) = \begin{cases} -1 & \text{if } y < 0 \\ 0 & \text{if } y = 0 \\ 1 & \text{if } y > 0 \end{cases}.$$

Boundary conditions along the faces of $\tilde{\Omega}^{0,1,0}$ are constructed similarly to (2.7).

Now consider the case of two Neumann boundary faces on $\partial\Omega$, say $x_1 = x_2 = 1$. By folding the domain first about the diagonal with endpoints at $(0, 1, x_3)$ and $(1, 0, x_3)$, then about the diagonal with endpoints at $(.5, .5, x_3)$ and $(1, 1, x_3)$, and finally about the x_3 axis, we are left with a wedge domain $\tilde{\Omega}^{1,1,0}$ with coordinates

$$\frac{1}{2} \leq x_1 \leq 1, \quad 1 - x_1 \leq x_2 \leq x_1, \quad \text{and} \quad \frac{1}{2} \leq x_3 \leq 1$$

FIG. 1. *Domain Reduction for the Cube*



(see Figure 3.1(c)). The operator $\mathcal{P}_{i_4, i_5, i_6}^{(1,1,0)}$ is given by

$$\mathcal{P}_{i_4, i_5, i_6}^{(1,1,0)} x = ((-1)^{i_4}(\text{sgn}(x_1) - x_1), (-1)^{i_5}(\text{sgn}(x_2) - x_2), (-1)^{i_6} x_3).$$

Boundary conditions along the faces of $\tilde{\Omega}^{1,1,0}$ are constructed similarly to (2.7).

Finally, consider the case of three Neumann boundary faces on $\partial\Omega$, say $x_1 = x_2 = x_3 = 1$. We fold the domain about two diagonals to get a tetrahedral domain $\bar{\Omega}^{1,1,1}$ with coordinates

$$0 \leq x_1 \leq 1, \quad 1 - x_1 \leq x_2 \leq x_1, \quad \text{and} \quad 1 - x_1 - x_2 \leq x_3 \leq 1$$

(see Figure 3.1(d)). This is a domain with one fourth the volume of Ω , but with only 45° and 90° angles. Let $y_i = (-1)^{i_4}(\text{sgn}(x_i) - x_i)$. The operator $\mathcal{P}_{i_4, i_5}^{(1,1,1)}$ is given by

$$\mathcal{P}_{i_4, i_5}^{(1,1,1)} x = ((-1)^{i_5}(\text{sgn}(y_1) - y_1), (-1)^{i_5}(\text{sgn}(y_2) - y_2), (-1)^{i_5}(\text{sgn}(x_3) - x_3).$$

If we allow 30° angles, then we can fold $\bar{\Omega}^{1,1,1}$ about the triangular plane with vertices $(1, 0, 1)$, $(.5, 1, .5)$, and $(1, 1, 1)$ into the tetrahedra domain $\hat{\Omega}^{1,1,1}$ with coordinates

$$\frac{1}{2} \leq x_1 \leq 1, \quad 2(1 - x_1) \leq x_2 \leq 1, \quad \text{and} \quad 2 - x_1 - x_2 \leq x_3 \leq x_1.$$

The operator $\mathcal{P}_{i_4, i_5, i_6}^{(1,1,1)}$ is constructed from $\mathcal{P}_{i_4, i_5}^{(1,1,1)}$. Boundary conditions along the faces of either $\bar{\Omega}^{1,1,1}$ or $\hat{\Omega}^{1,1,1}$ are constructed similarly to (2.7). Discretizing problems on $\tilde{\Omega}^{1,1,1}$ may present difficulties when central differences are involved, but do not for either finite elements or volumes.

Constructing subproblem matrices for the domains introduced in this section are, for the most part, rather straightforward. While constructing subproblem matrices for the domain $\hat{\Omega}^{1,1,1}$ may be painful for central difference discretizations, tetrahedra make sense when using a C^0 -piecewise linear basis functions in a finite element procedure.

4. Iterative Algorithm and Inexact Solvers. In practice, we often solve the subproblems (2.3) only approximately by some iterative method. In addition, the orthogonality relations $V_i \perp V_j$ in (2.2) may hold also only approximately. This is, for example, the case in the “robust multigrid method” by Hackbusch [11], where the subspaces are defined as ranges of suitable prolongation operators on a uniform grid. A convergence result for this method and comparisons with other prolongations operators for a model problem is in [10].

Define the energy norm by

$$\|u\| = \sqrt{a(u, u)},$$

and equip the space V with the inner product $a(u, v)$. The direct method of Section 2 now becomes the following iterative algorithm:

ALGORITHM 2. For an approximate solution u^m of (2.1), form the subproblems

$$(4.1) \quad \text{find } w_i \in V_i : a(w_i, v_i) = f(v_i) - a(u^m, v_i), \quad \forall v_i \in V_i,$$

and find approximate solutions $\bar{w}_i \in V_i$ such that

$$(4.2) \quad \|\bar{w}_i - w_i\| \leq \varepsilon \|w_i\|,$$

where $\varepsilon < 1$ is a constant. Then set

$$(4.3) \quad u^{m+1} = u^m + \sum_{i=1}^n \bar{w}_i.$$

Note that step (4.2) can be realized by solving (4.1) by an iterative method with convergence factor at worst ε in the energy norm, starting from an initial guess of zero. If more than one iterative method is used, then ε is the maximum of the convergence factors. If the subproblems are solved exactly, we get a method related to the well-known Schwarz alternating method (see [15]).

The following theorem gives the convergence factor of Algorithm 2 in terms of the (maximum) convergence factor of the iterative method (or methods) used to solve the subproblems and a measure of orthogonality of the subspaces V_i :

THEOREM 1. Let P_{V_i} be the orthogonal projection onto V_i , and λ_{max} and λ_{min} be the maximal and the minimal eigenvalues, respectively, of $\sum_{i=1}^n P_{V_i}$,

$$(4.4) \quad \lambda_{max} = \max \sigma \left(\sum_{i=1}^n P_{V_i} \right), \quad \lambda_{min} = \min \sigma \left(\sum_{i=1}^n P_{V_i} \right).$$

Then the iterates produced by Algorithm 2 satisfy the error bound

$$(4.5) \quad \|u^{m+1} - u\| \leq \left(\max\{\lambda_{max} - 1, 1 - \lambda_{min}\} + \varepsilon \lambda_{max} \right) \|u^m - u\|.$$

Proof. Consider the space $V_1 \times \cdots \times V_n$ with the inner product

$$\left((u_1, \dots, u_n), (v_1, \dots, v_n) \right)_{V_1 \times \cdots \times V_n} = \sum_{i=1}^n a(u_i, v_i)$$

and the corresponding norm $\|(u_1, \dots, u_n)\| = \left((u_1, \dots, u_n), (u_1, \dots, u_n) \right)^{1/2}$. Define the linear operator

$$A : V \rightarrow V_1 \times V_2 \times \cdots \times V_n, \quad A : v \rightarrow (P_1 v, \dots, P_n v),$$

which has the adjoint

$$A^* : V_1 \times V_2 \times \cdots \times V_n, \quad A^* : (v_1, v_2, \dots, v_n) \mapsto \sum_{i=1}^n v_i.$$

Then $A^* A = \sum_{i=1}^n P_i$ and so

$$(4.6) \quad \|A\| = \|A^*\| = \lambda_{max}^{1/2},$$

where the norm is defined in the usual operator sense. Now note that from (4.1) it follows that $w_i = P_i(u - u^m)$, and compute

$$\begin{aligned} u^{m+1} - u &= u^m - u + \sum_{i=1}^n \bar{w}_i = u^m - u + \sum_{i=1}^n w_i + \sum_{i=1}^n (\bar{w}_i - w_i) \\ &= \left(I - \sum_{i=1}^n P_i \right) (u^m - u) + A^*(\bar{w}_1 - w_1, \dots, \bar{w}_n - w_n). \end{aligned}$$

Consequently,

$$\|u^{m+1} - u\| \leq \left\| I - \sum_{i=1}^n P_i \right\| \|u^m - u\| + \|A^*\| \|(\bar{w}_1 - w_1, \dots, \bar{w}_n - w_n)\|$$

where by (4.2),

$$\begin{aligned} \|(\bar{w}_1 - w_1, \dots, \bar{w}_n - w_n)\| &\leq \varepsilon \|(P_1(u^m - u), \dots, P_n(u^m - u))\| = \varepsilon \|A(u^m - u)\| \\ &\leq \varepsilon \|A\| \|u^m - u\|, \end{aligned}$$

which concludes the proof using (4.6). \square

Note that we always have both $\lambda_{max} \geq 1$ and $0 < \lambda_{min} \leq 1$.

COROLLARY 2. *Using the same assumptions as in Theorem 1, if, in addition, $V_i \perp V_j$, for all i, j , then $\lambda_{max} = \lambda_{min} = 1$ and (4.5) reduces to*

$$\|u^{m+1} - u\| \leq \varepsilon \|u^m - u\|.$$

If the method converges with exact solvers (that is, for $\varepsilon = 0$), then $\lambda_{max} \leq 2$. In the case of two subspaces ($n = 2$), we know (see [14]) that

$$\lambda_{min} = 1 - \cos(V_1, V_2) \quad \text{and} \quad \lambda_{max} = 1 + \cos(V_1, V_2),$$

where the cosine of the two subspaces is defined in the inner product $a(u, v)$. In the general case ($n > 2$), we may well have $\lambda_{max} > 2$, and then Algorithm 2 diverges. Therefore, it is natural to consider one iteration of Algorithm 2 with a starting value of zero as an approximate solver for use as a preconditioner in the conjugate gradient method. The resulting linear operator should be linear and self-adjoint. So, we assume that for all $i = 1, \dots, n$, the approximate solvers are realized by linear iterative methods with starting initial guesses of zero and

$$(4.7) \quad \bar{w}_i - w_i = M_i w_i, \quad M_i : V_i \rightarrow V_i, \quad M_i^* = M_i.$$

Here, the adjoint is taken in the inner product $a(u, v)$ restricted to V_i . Choosing a basis in V , and identifying the elements of V and linear operators with their representations in that basis, we may write the result of such approximate solver in terms of matrices as

$$\tilde{u} = u^1 = C^{-1} f$$

and the original problem as $Au = f$. To apply conjugate gradients, we need that the matrix C is symmetric, positive definite, or, equivalently, that the mapping $C^{-1}A$ be

self-adjoint and positive definite relative to $a(u, v)$. Then (see [1, 12]) the convergence rate of the conjugate gradient method can be bounded in terms of the condition number

$$(4.8) \quad \kappa = \frac{\max \sigma(C^{-1}A)}{\min \sigma(C^{-1}A)}.$$

THEOREM 3. *Let (4.7) hold. Then the mapping $C^{-1}A$ is self-adjoint in the inner product $a(u, v)$. In addition, if λ_{max} and λ_{min} are given by (4.4) and all eigenvalues μ of the operators M_i satisfy*

$$-1 < \mu_{min} \leq \mu \leq \mu_{max} < 1,$$

then $C^{-1}A$ is positive definite and it holds for the condition number κ defined by (4.8) that

$$\kappa \leq \frac{(1 + \mu_{max})\lambda_{max}}{(1 + \mu_{min})\lambda_{min}}.$$

In particular, if $\|M_i\| \leq \varepsilon < 1$, then

$$\kappa \leq \frac{(1 + \varepsilon)\lambda_{max}}{(1 - \varepsilon)\lambda_{min}}.$$

Proof. Using (4.7), we get for the approximate solution $\tilde{u} = u^1$ obtained by one iteration of Algorithm 2 with $u^0 = 0$ that

$$\begin{aligned} \tilde{u} &= \sum_{i=1}^n \bar{w}_i = \sum_{i=1}^n w_i + \sum_{i=1}^n \bar{w}_i - w_i \\ &= \sum_{i=1}^n P_i u + \sum_{i=1}^n M_i P_i u = \sum_{i=1}^n P_i (I + M_i) P_i u, \end{aligned}$$

so

$$C^{-1}A = \sum_{i=1}^n P_i (I + M_i) P_i,$$

which is a self-adjoint operator relative to $a(u, v)$. The proof is concluded by considering the Rayleigh quotient

$$\frac{a(\sum_{i=1}^n P_i (I + M_i) P_i u, u)}{a(u, u)}$$

and noting that the extreme eigenvalues of $\sum_{i=1}^n P_i$ are obtained from the case when $M_i = 0$ for all i . \square

We note that trivially, we always have $\lambda_{max} \leq n$. A lower bound on λ_{min} follows from the following result (see [13, p. 7] or [16, Lemma 4]): if there is a constant $c_1 > 0$ such that for all $u \in V$, $u = \sum_{i=1}^n u_i$, $u_i \in V_i$, it holds that $c_1 \sum_{i=1}^n \|u_i\|^2 \leq \|u\|^2$, so $\lambda_{min} \geq c_1$. Similarly, if $c_2 \sum_{i=1}^n \|u_i\|^2 \geq \|u\|^2$ for all u , then $\lambda_{max} \leq c_2$ (cf., [2]).

Acknowledgements. We would like to thank Franco Brezzi for discussions about §3. The parallel computers mentioned in this paper belong to the Computer Science Department, Yale University, where the first author is a research affiliate. Part of this work was done while the first author was a visitor at the Center for Applied Mathematics, Purdue University, and also at the University of Colorado at Denver. Part of this work was done while the second author was a visitor at the IBM T. J. Watson Research Center.

REFERENCES

- [1] B. B. BELFORD AND J. H. E. KAUFMAN, *An application of approximation theory to an error estimate in linear algebra*, Math. Comp., 28 (1974), pp. 711–712.
- [2] P. E. BJØRSTADT AND J. MANDEL, *Spectra of sums of orthogonal projections with applications to parallel computing*. In preparation.
- [3] F. BREZZI, C. C. DOUGLAS, AND L. D. MARINI, *A parallel domain reduction method*, Numer. Meth. for PDE, 5 (1989), pp. 195–202.
- [4] N. CARRIERO AND D. GELERNTER, *The S/Net's Linda kernel*, ACM Trans. Comp. Sys., (1986).
- [5] C. C. DOUGLAS, *Madpack (version 2) users' guide*. Contact the author for availability information.
- [6] ———, *Multi-grid algorithms for elliptic boundary-value problems*, PhD thesis, Yale University, May 1982. Also, Computer Science Department, Yale University, Technical Report 223.
- [7] ———, *Multi-grid algorithms with applications to elliptic boundary-value problems*, SIAM J. Numer. Anal., 21 (1984), pp. 236–254.
- [8] C. C. DOUGLAS AND W. L. MIRANKER, *Constructive interference in parallel algorithms*, SIAM J. Numer. Anal., 25 (1988), pp. 376–398.
- [9] ———, *Some nontelegraphing parallel algorithms based on serial multigrid/aggregation/disaggregation techniques*, in Multigrid Methods: Theory, Applications, and Supercomputing, S. F. McCormick, ed., Marcel Dekker, New York, 1988, pp. 167–176.
- [10] C. C. DOUGLAS AND B. F. SMITH, *Using symmetries and antisymmetries to analyze a parallel multigrid algorithm*, SIAM J. Numer. Anal., 26 (1989), pp. 1439–1461.
- [11] W. HACKBUSCH, *A new approach to robust multi-grid solvers*, in ICIAM'87: Proceedings of the First International Conference on Industrial and Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia, 1988, pp. 114–126.
- [12] S. KANIEL, *Estimates for some computational techniques in linear algebra*, Math. Comp., (1966), pp. 369–378.
- [13] P. L. LIONS, *On the Schwarz alternating method I*, in Domain Decomposition Methods for Partial Differential Equations, R. Glowinski, G. H. Golub, G. A. Meurant, and J. Périaux, eds., Society for Industrial and Applied Mathematics, Philadelphia, 1988, pp. 1–42.
- [14] J. MANDEL AND S. F. MCCORMICK, *Iterative solution of elliptic equations with refinement: the model multi-level case*, in Domain Decomposition Methods for Partial Differential Equations II, T. Chan, R. Glowinski, G. A. Meurant, J. Périaux, and O. Widlund, eds., Society for Industrial and Applied Mathematics, Philadelphia, 1989, pp. 93–102.
- [15] H. A. SCHWARZ, *Über einige Abbildungsaufgaben*, Ges. Math. Abh., 11 (1869), pp. 65–83.
- [16] O. B. WIDLUND, *Optimal iterative refinement methods*, in Domain Decomposition Methods for Partial Differential Equations II, T. Chan, R. Glowinski, G. A. Meurant, J. Périaux, and O. Widlund, eds., Society for Industrial and Applied Mathematics, Philadelphia, 1989, pp. 114–126.